VB.NET
Programming Fundamentals
3rd edition

Ależandra B. Nicolas
Philip Erwin M. Ortiz
Jaime D.L. Caro, Ph.D.
FOREWORD

Everything is connected to everything else.

According to the chaos theory in mathematics and physics, it is possible for a butterfly to flap its wings at one end of the world and create a hurricane at the opposite end. (Hence the more popular term “butterfly effect.”) Similarly, the little things that we do may have an impact on society in ways that we can never expect.

Technology is a large factor in linking us to other people. Because of computers and the Internet, we are now connected to virtually everyone in the world. Furthermore, communication has never been quicker and more efficient. With the Information Superhighway, we have access to all sorts of data and gadgets. This would not have been possible a mere twenty years ago. Thus, we need to be aware that we can affect and influence others just by being able to send e-mails, design websites, create presentation reports, compile databases, or make audio/video files.

Our courseware is specifically designed to equip the students with the necessary knowledge and skills so they can navigate the terrain of present technology. Consider this book as a guide to strengthening human productivity and a tool for exploring the twenty-first century.

Jaime D.L. Caro, Ph.D.
Series Editor
ABOUT THE AUTHORS

Alezandra B. Nicolas is a Microsoft-certified trainer and solution developer for .NET. She creates Microsoft .NET applications for Windows using VB.NET. Her professional experience includes work as a technical specialist, consultant, and project manager for I.T. outsourcing and consulting companies. She has also taught different programming languages including VB.NET and ADO.NET. She teaches various courses at the UP Information Technology Training Center.

Philip Erwin M. Ortiz graduated from the University of Asia and the Pacific with a degree in BS Information Technology. He specializes in web and mobile applications development. His professional experience includes development under several platforms such as Visual Studio, Java, and a number of Web languages. He also provides consulting services for cloud computing systems development.

ABOUT THE SERIES EDITOR

Jaime D.L. Caro, Ph.D. has more than 20 years of experience in education and research in the areas of Computer Science, Information Technology, and Mathematics. He received the degrees of Bachelor of Science major in Mathematics (cum laude) in 1986, Master of Science in Mathematics in 1994, and Doctor of Philosophy in Mathematics in 1996, all from the University of the Philippines Diliman. He spent a year as a post doctorate research fellow at the University of Oxford from 1997 to 1998. He is presently Assistant Vice President for Development of the University of the Philippines, Program Director of the UP Information Technology Training Center (UP ITTC), and a professor of Computer Science in UP Diliman. He is an honorary member of the Philippine Society of Information Technology Educators (PSITE), President of the Computing Society of the Philippines (CSP), and a member of the Technical Panel on Information Technology Education of the Commission on Higher Education (CHED). Dr. Caro is a recognized expert on Complexity Theory, Combinatorial Network Theory, Online Communities, and e-Learning.
# TABLE OF CONTENTS

Lesson 1. An Introduction to Computer Programming 3
- Computer Programming
- The .NET Framework and Visual Basic
- Visual Studio 2010 IDE

Lesson 2. Front End Application 15
- Beginning a VB .NET Project
- Forms and Controls
- Combining Forms and Controls
- Changing Object Properties

Lesson 3. Back End Application 25
- Variable Data Types
- Declaring Variables
- The Event Procedure
- Merging the Front End and Back End Applications

Lesson 4. Working with Operators and Making Choices 35
- Arithmetic Operators
- Comparison Operators
- Logical Operators
- Concatenation Operators
- Decision Structure

Lesson 5. Looping and Arrays 49
- While Loop
- Do Loop
- For Loop
- Arrays
<table>
<thead>
<tr>
<th>Lesson</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Organizing Your Program</td>
<td>63</td>
</tr>
<tr>
<td></td>
<td>Procedure</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Subroutine</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Functions</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Custom Modules</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Access Modifiers</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Introduction to OOP in VB.NET</td>
<td>77</td>
</tr>
<tr>
<td></td>
<td>Object-Oriented Programming</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Understanding Classes and Objects</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Creating a Custom Class</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Properties and Methods for Custom Classes</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Using Custom Class Libraries</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>User Accessibility</td>
<td>89</td>
</tr>
<tr>
<td></td>
<td>Message Box</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Multiple Forms</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Dialog Box</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Getting Return Values from Other Forms</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Menu</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Accessing Data</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>Using Text Files</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Opening and Reading a Text File</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Manipulating Strings</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Writing to a Text File</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Using a Database</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ADO.NET Model</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Completing an Application</td>
<td>109</td>
</tr>
<tr>
<td></td>
<td>Software Development Life Cycle (SDLC)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The Development Process</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Wrapping Up Your Program</td>
<td></td>
</tr>
</tbody>
</table>
INTRODUCTION

Software applications, ranging from complex information management systems to games, are generally thought to be difficult to create. This usually discourages prospective students and professionals alike to dive into programming software applications. Seeing their plight, Microsoft released Visual Studio to let programmers develop applications “visually” through an integrated development environment (IDE) just by dragging and dropping common controls like textboxes and buttons from the toolbox. Included in the Visual Studio suite is Visual Basic, a powerful and easy-to-learn tool for creating Windows applications. Its pseudocode-like syntax makes it easy for users to grasp programming concepts like variable declaration and flow control.

This courseware teaches the basics of programming using Visual Basic .NET, such as declaring and using variables, performing arithmetic and logical operations, looping, and arrays. Advanced topics like the .NET framework, object-oriented programming (OOP), and data-driven applications are introduced and integrated early on to allow the students to appreciate the finer points of current programming techniques and code reusability.

LEARNING GOALS

By the end of this courseware, the student is expected to:

1. Understand VB.NET’s programming syntax and IDE.
2. Apply OOP and .NET framework principles to solve simple programming problems.
3. Create visually appealing and data-driven applications.
“I was reflecting, in the first place,” replied Dantes, “upon the enormous degree of intelligence and ability you must have employed to reach the high perfection to which you have attained. What would you not have accomplished if you had been free?”

“Possibly nothing at all; the overflow of my brain would probably, in a state of freedom, have evaporated in a thousand follies; misfortune is needed to bring to light the treasures of the human intellect. Compression is needed to explode gunpowder. Captivity has brought my mental faculties to a focus; and you are well aware that from the collision of clouds electricity is produced — from electricity, lightning, from lightning, illumination.”

—The Count of Monte Cristo, by Alexandre Dumas
In the story, Dantes expresses his awe at the wise old man and imagines (or tries to imagine) all the marvelous things this man can do if he was not in prison. For the sake of argument, the man says that freedom would probably overwhelm his intellectual faculty. How about you? What do you think about the power of the human brain? Just look at all our accomplishments throughout the centuries. We enjoy a level of technology today that allows for much of the things that we take for granted, such as lights produced by the artificial conduction of electricity, images and sound transmitted through the air, and algorithms translated into software to run a humanoid robot—all of these feats are part of our evolution as thinking beings.

Computer Programming

Some people have very broad ideas about computer programming. Most of these stem from science fiction movies such as those where machines imbued with artificial intelligence rise up against mankind (Terminator) and those where computers act as friends (A.I.: Artificial Intelligence) helping people with their tasks and sometimes even giving guidance. Perhaps one day, machines may indeed develop that degree of intelligence; for the time being, however, computers depend on humans for instructions.

So what can a computer actually do? For one thing, a computer can perform calculations faster than any person can, and perform these accurately as well. It can also store large amounts of information and retrieve them quickly.

Speed, accuracy, and massive amounts of information storage characterize computers, but these attributes are dependent on the step-by-step instructions given to them. In general, giving instructions in sequence to computers is called programming, and the instructions are referred to as a computer program. Computers depend on programs to perform what humans want them to do.
A very good example of a computer program is the Microsoft Windows operating system (OS). It is a program that provides a visual interface for interaction with a computer as an operating system runs in the background and manages the usage of the computer’s hardware and all other computer programs that are open. Another example of a computer program is the word processing program of Microsoft Office and OpenOffice. Computer programs are also used for entertainment—from watching movies to playing an online role playing game (RPG).

Computer programs are actually at work around us even if there’s no one around using a personal computer. For example, programs run in point-of-sale (POS) vending machines every time someone makes a purchase from them; it’s also programs that serve as the functional infrastructure of money management systems that keep track of people’s hard earned money. It’s a fact that even the traffic lights on the street in front of banks use computer programs. Computer programs are simply everywhere these days.

Every time and anywhere we use a computer, we come across various computer programs. Computer programs have become a staple in today’s world and practically any organization has a need for one, be it off-the-rack or custom-made. Imagine the trouble a non-profit organization has to go through if the staff must pore over membership information from tons of stacks and files of paper! The task is not a problem if only a handful of members are involved; but what if you have a million members? How would you feel if you were the employee searching for information in stuffy files? Getting information manually is tedious, so that is one work area where a computer program becomes indispensable. Computer programs are practical solutions to enable more effective and efficient work.

Generations of Programming Languages

Computers are machines and they can only understand instructions which are in machine language. It’s the same way with human languages. If you try to talk Swahili to someone who speaks only German, then you will not be able to communicate properly. He will hear you but will not be able to reply or respond to you properly. To make yourself understood, you must use a more basic and primal way of communication, like sign language, which is sort of like the first generation language of human beings. Fundamental human language has diversified and evolved through the ages, but people from different cultures and walks of life are still able to understand each other using simpler forms of communication. Computer language has paralleled human language evolution in that it has branched out into the many forms in existence today. Yet the computer can only be made to understand them all if they are translated first into a base language.

The evolution of programming languages can be categorized into generations.

First Generation (1GL)

Computers are essentially machines that consist of switches with an electrical current passing through them. The actions of a computer are determined by a specific sequence of switches that are either “on” (electricity flowing through), or “off” (no electrical flow present). For programmers to visualize a sequence, a switch that is on is represented as “1” and the switch that is off, by “0”. This logic is the basis of machine language which is essentially just a series of ones and zeroes. In order to
control a computer back in the old days, people relied on switches in the front panel of a computer that were flipped on or off. It’s a system that works in theory, but one which is too tedious in actual practice for any real work to be done.

**Second Generation (2GL)**

*Assembly language* is known as low-level programming language. Instead of ones and zeroes defined by switches, its programmers have a specific set of mnemonics which can be translated by a program to machine language. Just like its 1GL predecessors, assembly language is specific to a certain processor and therefore cannot be ported to different machines. For many programmers, assembly language is difficult to understand and thus, solutions were created to allow them to communicate with computers like how they would talk to another person.

**Third Generation (3GL)**

This type of language is normally categorized as high-level. Whenever we hear the word “programming,” examples of 3GL come to mind. COBOL was introduced in the 1950s, and is one of the early examples of 3GL which is still being used until today. Other more popular examples of 3GL are C++, Java, and Visual Basic.

**Translation of Programming Languages**

Whatever programming language you chose, computers will still need to receive the instructions in a format that they can understand, hence the need for the conversion of the high-level programming language into machine language format. The translation is now done using a computer program. There are generally two types of programs that perform translation. An **interpreter** is a program that translates high-level language instructions into machine code line by line as the program is being run. A **compiler**, on the other hand, translates high-level instructions into machine code before running the program.

**The .NET Framework and Visual Basic**

.NET (Dotnet) and Visual Basic were created to enhance and complement the Microsoft Windows operating system. Let us now take a brief trip along memory lane to see how it all started.

**The History of Computers**

There was a time when computers took up whole rooms and needed a lot of electricity to run. Not only were computers expensive to buy and maintain, but they were also very difficult to program. It was around 1975 when an ad for a computer targeting mainstream computer enthusiasts was published. The announcement of the Altair System rocked the world of businessmen, and some of them, like Bill Gates and Paul Allen, knew that it was the start of something big. They knew that the future of computing lay in software.
Fueled by the desire to make computers more accessible to the general public, Gates and Allen developed Microsoft BASIC for the Altair system. It was the first computer language program written for a computer system. Later, Gates and Allen would form Microsoft, which was initially recognized not for the Windows operating system, but for creating and integrating programming languages for various computer systems. It was the IBM contract with Microsoft which led to the development of an operating system which allowed computers to become more personal.

Having written several versions of BASIC for various computer systems, Microsoft was more than happy to embark on this new challenge. It was in 1981 that IBM launched the Personal Computer running under the Microsoft Disk Operating System (MS DOS). It was then that Gates's vision of having a computer in every home had begun to truly take shape. Having a solid track record for creating computer programming languages, Microsoft expanded to operating systems. MS DOS later on evolved to the popular Windows Operating System which we all know today. As Microsoft's operating system grew, so did the number of bugs that hounded users and programmers alike. Microsoft's competition came from all fronts. The Sun Microsystems Java programming language was a rising star in the 1990s, and the free operating system, Linux, also loomed in the background.

To remain competitive, Microsoft needed to address the issues plaguing its operating system, as well as to beef up their programming languages, for the true life blood of an operating system was the number of programming applications that ran on it.

Microsoft found its solution in the .NET Framework.

.NET Framework

The .NET (pronounced as dot net) framework is the core of the .NET infrastructure, which is a collection of services and classes designed to help simplify application development. It is a platform for building and running applications. One of the main goals of this framework is to enable programmers to develop interoperable applications for any device or platform, but it can also be used to build traditional applications that will run on the Windows desktop. The .NET framework is not a programming platform or a programming language. It is a general infrastructure for the use of the Windows Operating System and all other programs running on it.

Some of the key benefits of .NET are as follows:

Consistent Programming Models

Programming languages have different ways to do the same thing. With .NET, programming languages for the Visual Studio platform, such as Visual Basic, Visual C++, and Visual C#, can use
the same classes to do a certain task. For example, the classes used to connect to a database or classes used to actually display the text on the screen can be shared across different programming languages through the .NET class library. This means that accessing a database using Visual Basic and C# will more or less be the same, apart from syntactic elements.

Security Enhancement

If you are making a program that will run on just one machine, security is not that much of a concern. But if your program works across several computers or connects with other programs, security becomes a factor to consider. The programmer has to make sure that his program will not cause problems with other programs and should be more or less safe from malicious programs. With .NET, developers and systems administrators can set security levels for applications.

Simplified Development

With .NET, it is possible to write presentation logic (the front-end of an application) separately from the application logic (the back end code). The .NET framework also aids in tracking down the bugs of your program through a runtime diagnostic.

Easier Application Deployment

After running an installer or creating a folder with the executable version of the code, .NET takes care of locating and loading the components needed for the application. .NET also checks if all the components needed by the program are available on the computer before the program is executed.

Microsoft Visual Studio

Visual Studio is an integrated development environment (IDE) that is used to develop computer programs running under the Microsoft Windows environment such as the Windows Operating System, Windows Mobile, and Microsoft Silverlight. Being an IDE, it comes with four components designed for ease of programming development. These components are:

- Source Code Editor
- Compilers and Interpreters
- Build Automation Tools
- Error Debugger

One of the most common misconceptions about Microsoft programming languages or Visual Studio is that the products can be very expensive. What most people don’t realize is that the Visual Studio platform has several editions—from the completely free to expensive systems that allow teams of programmers to work together. Applications made using the free Visual Studio Express Editions can also be used for commercial use without licensing restrictions.

In this book, we shall be using Visual Basic 2010 Express which can be downloaded for free in the Microsoft website at http://www.microsoft.com/express/Downloads/.
**Microsoft Visual Basic**

Visual Basic started as a drag-and-drop shell prototype named Tripod developed by Alan Cooper. The history of Visual Basic is tied with the evolution of operating systems. Before the introduction of the graphical user interface (GUI) on operating systems, the only thing that appeared on a computer screen was plain text.

As operating systems became more complex and started to use the GUI, programming languages also had to evolve to enable developers to create programs that were graphically rich as against simple text programs running on a command prompt.

Microsoft bought Tripod and combined it with the programming language BASIC to create a programming system that can use Windows forms to run in its operating systems. BASIC was developed by Dr. John G. Kenney and Thomas Kurts in 1964 and stands for Beginner's All-Purpose Symbolic Instruction Code. It was intended to be a simple language for teaching students programming and had very limited memory and processing capability. Bill Gates and Paul Allen eventually expanded BASIC to be incorporated to various computer systems used at the time.

Visual Basic 1.0 was released in 1991 but it was in 1992, with Visual Basic 2.0, that the popularity of the programming language skyrocketed. Through the years, there have been many changes in Visual Basic, and today it is one of the more popular programming languages capable of building rich Windows-based form applications for general use.

**Visual Studio 2010 IDE**

The following figure is a window similar to what is shown to you when you start to create a new Windows application project in Visual Studio .NET.
The main window has several standard interface elements found in the Windows environment, such as the toolbar and the menu bar. On top of these standard controls, the Visual Studio .NET IDE contains additional user interface elements:

**Windows Form Designer** - This is the canvas of the Visual Basic application. You can drag and drop controls from the Toolbox to any part of the form area and position these controls wherever you see fit. The Form Designer does not just allow you to see your application graphically but also enables you to see the programming code behind the form by clicking on F7 or by doing a right mouse click on the Form Designer Area and then selecting View Code. The workspace in the Windows Form Designer can load tabs for the code and design sections of your project, which should save some time and confusion when you’re knee-deep in your project.

**Solution Explorer Window** - This lists the solution name, project name, and all forms and modules used in the project. You can open a particular file existing in a project by double-clicking the file in this window. You can also click on the View Code and View Design icons and see its corresponding window pop-up in the Windows Form Designer area.

**Properties Window** - This displays the properties associated with an object. It lists all the properties associated with forms such as Name, Size, Font, Text, etc.

**Toolbox** - This houses several tabs that represent the controls that can be used in your application.

**Data Sources Window** - The Microsoft Developer Network (MSDN) describes the data sources window as a window “that displays the data sources in your project. Data sources represent the data available to your application. You can use the Data Sources window to create data-bound controls in your user interface by dragging items from the window onto a design surface in your project.”

**Error List** - This is perhaps the most helpful window for developers since it shows the messages or errors that caused the application to fail after it has been compiled.

**NOTE**

You can move the windows and panels of the Visual Basic IDE around the screen to any place you want. Just click on the panel you want to move, hold down the left mouse button and drag the panel to the location you want. You can also pin the panels to a certain location.
SUMMARY

There are lots of programming languages available, with the most popular ones being classified within the third generation of programming languages. Being a programmer or a developer does not stop with writing a code. The Visual Studio platform is considered as one of the most popular development platforms due to the advantages brought about by the .NET Framework. Visual Basic is one of the languages available in Visual Studio 2010 and shall be the focus of this book.

WORD BANK

.NET Framework – defines the environment that you use to execute Visual Basic .NET applications and other services that you can use within those applications.

Syntax – grammatical rules and patterns on how to write a code specific to a certain programming language.

Visual Studio .NET – an integrated environment that allows you to easily design, develop, debug, and deploy .NET Framework solutions.
True or False. Write TRUE if the given statement states fact, otherwise write FALSE.

1. ________ A compiler translates high-level languages line by line to machine language.
2. ________ In layman’s terms, 3GL is normally associated with the word “programming.”
3. ________ GUI stands for graphical user interface.
4. ________ The Error List shows the errors that caused the application to fail after it has been compiled.
5. ________ By default, Visual Studio can be run by going to the Start menu, then All Programs, and then Microsoft Visual Studio .NET.
6. ________ A computer cannot perform calculations faster than a person can.
7. ________ Giving step-by-step instructions to computers is called programming.
8. ________ The instructions written by a programmer is called a program.
9. ________ Machine language exists as a series of 1s and 2s.
10. ________ High-level languages are translated line by line to machine language by an interpreter.
11. ________ BASIC stands for “Beginner’s All-Purpose Symbolic Instruction Category.”
12. ________ The front-end of an application is known as the application logic.
13. ________ Computer programs can only be seen on a computer.
14. ________ The .NET Framework is a programming language environment.
15. ________ Computers have always targeted the mainstream consumer.
SKILLS WORKOUT

Write your answer on the blanks provided.

1. Explain how computer programs impact modern society.

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

2. Why is the .NET Framework beneficial when it comes to developing computer programs?

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

3. Enumerate and describe the elements of a Visual Studio IDE.

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________
GROUP WORK

The focus of this book is Visual Basic .NET, but it’s always good to know the other kinds of languages available. A good programmer is ideally proficient with several programming languages. The class will be divided into three groups. Each group should list down programming languages (other than Visual Basic) that they have heard of. The group will do an analysis and rate each one from 1 to 5 (1 being the highest), to determine which is the most useful for programmers. The group should then present its report in class.
THIS IS A SAMPLE COPY

To purchase the full version of this book, go to www.techfactorsinc.com/store